Recovery of disrupted airline operations using k-Maximum Matching in graphs

Julien Bensmail[1], Valentin Garnero[1], Nicolas Nisse[1], Alexandre Salch[2] and Valentin Weber[2]

[1] *Université Côte d'Azur, CNRS, Inria, I3S, France*
[2] *Innovation & Research, Amadeus IT*

Abstract

By Berge's theorem, finding a *maximum matching* in a graph relies on *the use* of *augmenting paths*. When no further constraint is added, Edmonds' algorithm allows to compute a maximum matching in polynomial time by sequentially augmenting such paths. Motivated by applications in the scheduling of airline operations, we consider a similar problem where only paths of bounded length can be augmented. Precisely, let $k \geq 1$ be an odd integer, a graph G and a matching M of G. What is the maximum size of a matching that can be obtained from M by *using* only augmenting paths of length *at most* k?

We first prove that this problem can be solved in polynomial time for $k \leq 3$ in any graph and that it is NP-complete for any fixed $k \geq 5$ in the class of planar bipartite graphs of degree at most 3 and arbitrarily large girth. We then prove that this problem is in P, for any k, in several subclasses of trees such as caterpillars or trees with all vertices of degree at least 3 "far appart". Moreover, this problem can be solved in time $O(n)$ in the class of n-node trees when k and the maximum degree are fixed parameters. Finally, we consider a more constrained problem where only paths of length *exactly* k can be augmented. We prove that this latter problem becomes NP-complete for any fixed $k \geq 3$ and in trees when k is part of the input.

Keywords: Graph; Matching; Augmenting paths; Complexity; Trees.

1 Recovery of disrupted airline operations

For obvious safety reasons, the airport authorities should ensure that the number of landing aircraft in a given time slot is less than the capacity of the airport. Thus, airports have time intervals (*slots*) that are initially assigned to aircraft according to their arrival schedule. An aircraft can only be assigned to a slot that is compatible with its arrival time. However, if a flight is delayed, airline operation controllers must assign a new slot via the information system managing these exchanges. The strict regulations that respects this system are that only two operations are possible [6]. Either an available slot is assigned to the aircraft A (**Rule** 1), or a slot S which is already assigned to another aircraft B is reassigned to A while B is assigned an available slot S' (**Rule** 2). In both cases, the slot S must be compatible with the schedule of A and in the second case, S and S' must be compatible with the one of B. If several planes fall behind and lose their slots, the resolution of these problems is difficult for airline operation controllers [8] that do not have the tools to make these changes and must ensure "by hand" that all aircraft are going to get a slot.

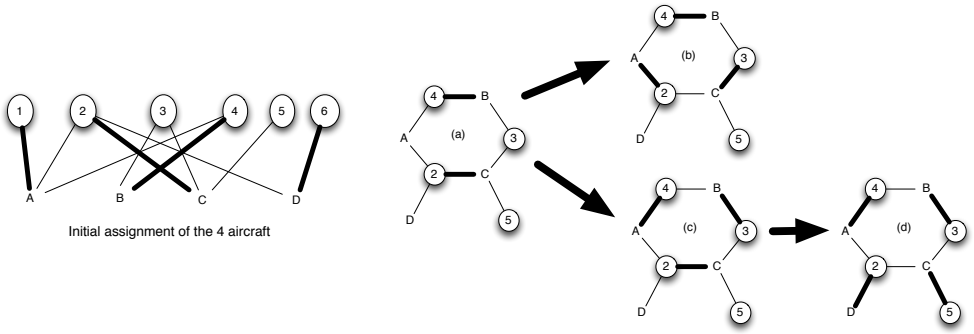

Fig. 1. Example of a simple scenario with 4 aircraft and 6 slots.

An example of a simple scenario with four planes A, B, C, D and 6 slots is shown in Fig. 1. Each aircraft is linked to the slots that are compatible with it (left). Initially, Slots $1, 2, 4$ and 6 are assigned to A, C, B and D, respectively. Edges of the matching are depicted in bold. Say that, after A and D have been delayed, they are not compatible with slots 1 and 6 anymore. Hence, the configuration becomes the one depicted in Fig. 1(a). From Configuration (a), if Slot 2 is reassigned to A and Slot 3 is assigned to C (i.e., Rule 2 is applied to A and C), then we reach Conf.(b) where no allowed modification is possible (no Rule can be applied). Another solution would be to apply Rule 2 to A and B (A gets Slot 4 and B gets Slot 3), reaching Conf.(c), and then to apply Rule 2 to C and D (C gets Slot 5 and D gets Slot 2), reaching the Conf.(d) where all aircraft are assigned a slot.

Matching in graphs. The problem of *reassignment of slots* can of course be modeled as a problem of matching in graphs. A *matching* $M \subseteq E$ of a graph $G = (V, E)$ is a set of edges pairwise disjoint. The maximum size of a matching in G is denoted by $\mu(G)$. The problem of computing a maximum matching has been widely studied and it is well known that it can be solved in polynomial time [2,4]. A key ingredient in most of the work on matchings is the notion of *augmenting path*. A path $P = (v_0, \cdots, v_k)$ of G is a sequence of pairwise distinct vertices such that $e_i = \{v_i, v_{i+1}\} \in E$ for each $0 \leq i < k$. The path P is said *M-augmenting* if v_0 and v_k are exposed (not incident to an edge of M) and, for any $0 \leq i < k$, $e_i \in M$ if and only if i is odd. A well known theorem of Berge states that a matching M is maximum if and only if there are no M-augmenting paths [1]. In particular, if P is M-augmenting, then $M \Delta E(P)$ is a matching of G with size $|M| + 1$, where $E(P)$ is the set of edges of P and Δ is the symmetric difference. When passing from a matching M to the matching $M \Delta E(P)$, we say that the M-augmenting path P is *augmented*. Some algorithms that compute maximum matchings in graphs are based on augmenting such paths in the non-decreasing order of their lengths [3,5]. For instance, augmenting only the paths of length at most $2k-3$ provides a $(1-1/k)$-approximation of the maximum matching [3]. The problem of augmenting a matching with bounded-length paths has also been studied in the context of wireless networks, e.g., for computing scheduling of transmissions with interference [9].

The problem of *reassignment of slots* described above can be modeled as follows. Let $G = (X \cup Y, E)$ be a bipartite graph. X represents the set of aircraft and Y represents the set of slots. There is an edge between $a \in X$ and $s \in Y$ if the slot s is compatible with the schedule of the aircraft a. Let M be a matching of G that corresponds to a pre-established valid assignment of some slots to some aircraft (see Fig 1(a)). The problem of reassignment of slots is equivalent to computing a maximum matching that can be obtained from M by augmenting only paths of length at most 3. For instance, the first scenario in the above example consists in augmenting the path $(A2C3)$. Reaching Conf.(b), there are no augmenting paths of length at most 3. The second scenario consists in first augmenting the path $(A4B3)$ (reaching Conf.(c)) and then the path $(D2C5)$ (reaching Conf.(d)). This example already suggests that bounding the length of augmented paths may increase the difficulty of the Maximum Matching Problem since the order in which the paths are augmented becomes important.

Our results[1] Let k be an odd integer. We consider the problem that takes a graph $G = (V, E)$ and a matching $M \subseteq E$ as inputs. Let $\mu_{\leq k}(G, M)$ denote the size of a maximum matching that can be obtained from M in G by augmenting paths of length at most k. The *k-Matching Problem* (denoted by $MP^{\leq k}$) consists in computing a sequence of augmenting paths of length at most k that allow, starting from M, to obtain a matching of size $\mu_{\leq k}(G, M)$.

In the cases $k \in \{1, 3\}$, we prove that the computational complexity of the k-Matching Problem is equivalent to the one of the classical maximum matching problem (without any constraint on the length of paths). Hence, for $k \in \{1, 3\}$, it can be solved in polynomial time. Then, we show that, for any odd integer $k \geq 5$, the problem is NP-complete in planar bipartite graphs with maximum degree at most 3 and arbitrarily large girth. We also prove that the problem can be solved in polynomial time, for any k, in several subclasses of trees such as caterpillar and trees with all vertices of degree ≥ 3 "far appart". Moreover, for any fixed k, there is an $O(n)$-time algorithm for computing $\mu_{\leq k}(T, M)$ in trees with bounded maximum degree.

Toward understanding the complexity of our problem in trees, we also consider the more constrained problem $MP^{=k}$ where only paths of length *exactly* k can be augmented. Let $\mu_{=k}(G, M)$ denote the size of a maximum matching that can be obtained from M in G by augmenting paths of length exactly k. We prove that computing $\mu_{=k}(G, M)$ is NP-complete already for $k = 3$ and that it is NP-complete in trees when k is part of the input.

2 Constrained Maximum Matching Problems

In this section, we present our results and give brief intuitions of their proofs.

When $k \geq 3$, the difficulty arises from the fact that the order in which the paths are augmented is important. This fact is illustrated in Fig. 1 where augmenting first the path $(A2C3)$ leads to a non-optimal dead end configuration. In addition, the order in which the paths are augmented has an impact on the creation or non-creation of new augmenting paths of length at most k. For instance, for $k = 5$, let us consider the graph that consists of a path $(v_1, \cdots, v_7)$ plus three edges $\{v_5, v_8\}, \{v_8, v_9\}$ and $\{v_9, v_{10}\}$. The initial matching is $\{\{v_2, v_3\}, \{v_4, v_5\}, \{v_8, v_9\}\}$. Initially, there are two augmenting paths of length at most 5: $P_1 = (v_6, v_7)$ et $P_2 = (v_1, \cdots, v_6)$. If P_1 is augmented first, then there remain no augmenting paths of length at most 5 anymore. However, while augmenting P_2 destroys the path P_1, it also creates a new augmenting path $P_3 = (v_{10}, v_9, v_8, v_5, v_6, v_7)$ that can be augmented (cf., Fig. 2).

[1] Due to lack of space, the proofs are omitted and can be found in [7] and here: http://www-sop.inria.fr/members/Nicolas.Nisse/publications/matchingLAGOS.pdf.

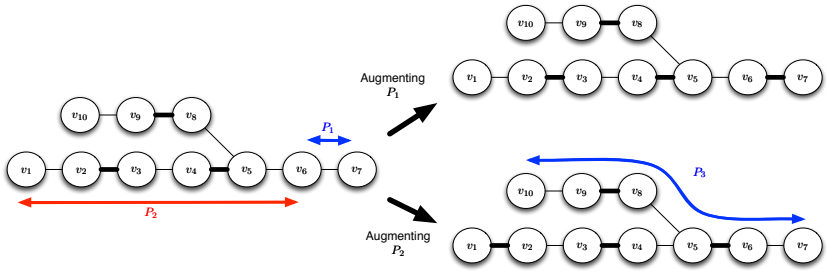

Fig. 2. Example of an instance (with $k = 5$) where augmenting a path (P_2) creates a new augmenting path (P_3) that must be augmented to reach an optimal solution. Matchings are represented by bold edges.

Theorem 2.1 *For any fixed odd $k \geq 5$ (resp., ≥ 3), $MP^{\leq k}$ (resp., $MP^{=k}$) is NP-complete in the class of planar bipartite graphs with degree at most 3 and arbitrarily large girth.*

Intuition. We do a reduction from planar 3-SAT. Intuitively, the fact that the order of the augmentations has an impact on the solution allows us to simulate the choices of the variables' assignments. Depending on the choices, some extra augmenting paths may be created and augmented to achieve a larger solution. Augmenting such a path corresponds to satisfying a clause. ⋄

For $MP^{\leq k}$, the main difference between the cases $k = 3$ and $k \geq 5$ is that, for $k = 3$, it is possible to "ignore" the augmenting paths that were not present in the initial matching M. That is, if a new augmenting path P of length at most k is created after augmenting some path, then it is never necessary to augment P in order to achieve a matching of maximum size $\mu_{\leq 3}(G, M)$.

Theorem 2.2 *For $k \in \{1, 3\}$, $MP^{\leq k}$ is in P in general graphs.*

Intuition. The case $k = 1$ is trivial. For $k = 3$, we prove that we can consider only the set $\mathcal{S}$ of augmenting paths of length ≤ 3 that are compatible with the initial matching. The algorithm first computes a classical maximum matching M^* (e.g., using Edmonds' algorithm) and then augments only the paths in $\mathcal{S}$ to achieve the matching M^*. The latter phase is possible because we prove that the paths in $\mathcal{S}$ are disjoint and do not interfere one with each other. ⋄

Note that Th. 2.2 solves our motivating problem about airline operations. Given our hardness results, we then naturally consider the class of trees.

Theorem 2.3 *For any odd k, $MP^{\leq k}$ and $MP^{=k}$ are in P in the classes of caterpillars and of trees where all vertices with degree ≥ 3 are at distance $> k$.*

Intuition. In the case of a caterpillar (a tree with a dominating path), we prove that there always exists an optimal solution augmenting "from left to right"

the augmenting paths of length $\leq k$ (or $= k$) initially present. Then, in trees T with a unique vertex c with degree ≥ 3, we prove that an optimal solution can be obtained by 1) performing a particular sequence of augmentations around c (which can be found in polynomial time), and 2) applying the caterpillar algorithm along the branches. Finally, for trees with "far appart" vertices of degree ≥ 3, we propose an algorithm that considers locally the subtrees around high-degree vertices and then combines the solutions. $\diamond$

Theorem 2.4 *For any fixed odd k, $MP^{\leq k}$ and $MP^{=k}$ can be solved in linear time in the class of trees of bounded maximum degree Δ.*

Intuition. The algorithm proceeds by dynamic programming from the leaves to some root. For every vertex v, and for any sequence $\mathcal{S}$ of paths of length $\leq k$ (or $= k$) passing through v (the number of such sequences is bounded by a function of k and Δ), we compute a maximum solution for the subtree T_v rooted in v that augments the paths in $\mathcal{S}$. This can be done in constant time (depending on k and Δ) from the solutions for the children of v. $\diamond$

Finally, by a reduction from 3-SAT, we prove that:

Theorem 2.5 *$MP^{=k}$ is NP-complete in trees when k is part of the input.*

3 Further Work

The main open question concerns the complexity of $MP^{\leq k}$ in trees. Moreover, can $MP^{=k}$ be solved in polynomial-time (or FPT) in trees when k is a fixed parameter? The complexity of these problems in other graph classes (e.g., bounded treewidth graphs, interval graphs...) is also of interest.

References

[1] C. Berge, *Two theorems in graph theory*, Proc. of National Academy of Sciences of the USA **43(9)** (1957), 842–844.

[2] J. Edmonds, *Paths, trees, and flowers*, Canadian J. of Mathematics **17** (1965), 449-467.

[3] J. E. Hopcroft and R. M. Karp, *An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs*, SIAM J. Comput. **2(4)** (1973),225–231.

[4] H. W. Kuhn, *The Hungarian Method for the assignment problem*, Naval Research Logistics Quarterly **2** (1955), 83-97.

[5] S. Micali, V.V. Vazirani, *An $O(\sqrt{|V|}|E|)$ Algorithm for Finding Maximum Matching in General Graphs*, 21st Symp. on Foundations of Comp. Sc. (FOCS) (1980), 17-27.

[6] G. Sieg, *Grandfather rights in the market for airport slots*, Transportation Research Part B: Methodological **44(1)** (2010), 29-37.

[7] N. Nisse, A. Salch, V. Weber, *Recovery of disrupted airline operations using k-Maximum Matching in Graphs*. Research Report, INRIA-RR-8679, 2015.

[8] M. Wambsganss, *Collaborative decision making through dynamic information transfer*, Air Traffic Control Quarterly **4(2)** (1996), 109-125.

[9] X. Wu and R. Srikant, *Regulated maximal matching: A distributed scheduling algorithm for multi-hop wireless networks with node exclusive spectrum sharing*, IEEE Conf. on Decision and Control (2005).